Mar. 2004, Vol.19, No.2, pp.150–157 *J. Comput. Sci. & Technol.*

Performance Enhancement and Bandwidth Guarantee in IEEE 802.11 Wireless LANs

Yong Peng and Shi-Duan Cheng

National Laboratory of Switching Technology and Telecommunication Networks, Beijing University of Posts and Telecommunications, Beijing 100876, P.R. China

E-mail: wujing@bupt.edu.cn

Received November 13, 2002; revised June 17, 2003.

Abstract This paper first revisits the previously proposed NSAD (New Self-Adapt DCF) mechanism. Some modifications are presented to further enhance the performance of NSAD in the error-prone environment. Then a new MAC mechanism is proposed that can realize bandwidth guarantee by assigning different self-adapt parameters to users at different priority levels. The bandwidth guarantee property of this new mechanism is analyzed and the high priority users are found to have bandwidth guaranteed even in heavy contention condition, which is proved true not only by theoretical analysis but also by simulation results. At the same time the new scheme keeps the self-adapt character of NSAD, so the overall system utilization is kept very high in heavy contention condition compared with the previously studied DCF-based QoS mechanisms.

Keywords IEEE 802.11, performance enhancement, bandwidth guarantee

1 Introduction

More and more people have realized that it is really a convenient and cost-efficient way to gain access to the Internet through wireless LANs. As an important local access method of the 4G wireless communication system, wireless LAN has become a research focus among the researchers all over the world. IEEE 802.11 [1] protocol has gradually become the main wireless LAN standard since its birth in 1999 and its DCF mode is the most frequently used MAC algorithm presently. It can provide users with two different kinds of access methods, DCF (Distributed Coordination Function) and PCF (Point Coordination Function). DCF is the basic access method of IEEE 802.11 and can be used to construct the infrastructure networks and ad hoc type networks. The DCF access mode realizes asynchronous multi-access by using the contention window exponential backoff algorithm. Recent research has shown that the traditional DCF access mode will deteriorate throughput and fairness, especially when the number of nodes is large[2]. Also the IEEE 802.11 protocol is found inefficient in the case of multi-hop ad hoc communication topology, especially when upper transport layer is TCP[3]. And another drawback of original DCF access mode is its lacking of QoS guarantee.

Researchers have presented different methods for the performance enhancement and QoS guarantee in 802.11 wireless LANs. Imad Aad discussed in [3] the QoS provision of 802.11 DCF by assigning different MAC parameters (e.g., backoff increase function, InterFrame Space, Maximum frame length) to users at different QoS levels. Andras Veres presented in [4] the concept of Virtual MAC (VMAC) and Virtual Source (VS). VS uses VMAC to predict the usable service that can be provided to the users. In [6], the authors placed emphasis on bandwidth guarantee in a distributed manner by tuning contention window. Similar to 802.11e protocol (draft 3.0), they did not consider how to tune the transmission probability in heavy contention environment.

In this paper, we first introduce some improvements of our previously proposed mechanism *NSAD* (New Self-Adapt DCF algorithm)[7], which will be the basis of our later discussion. *NSAD* tunes the station initial contention window to achieve throughput limit without knowing the exact number of contending mobile stations. Based on *NSAD*, we propose a new mechanism of bandwidth guarantee in wireless LAN environment: *QoS-NSAD*. *QoS-NSAD* can provide high priority users with guaranteed bandwidth while keeping the overall system utilization at a high level. We set up

---

*Correspondence

This work is supported by the National Natural Science Foundation of China under Grant No.90204003, the National High Technology Development 863 Program of China under Grant No.2001AA112071, No.2001AA121052 and No.2002AA103063, and the Research Fund for the Doctoral Program of Higher Education (RFDP) of China under Grant No.20010013003.

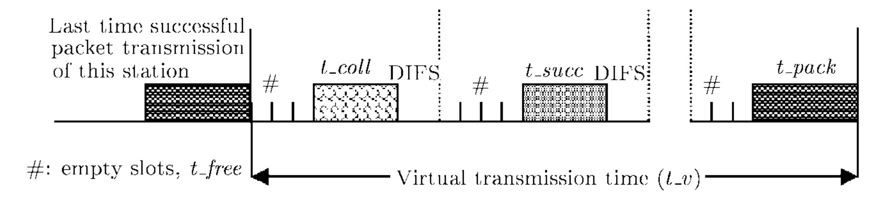


Fig.1. Data packet transmission process.

a new approximate model in order to theoretically analyze the performance of the *QoS-NSAD* protocol. We find that *QoS-NSAD* keeps the self-adapt character of *NSAD* and can tune the access probability of stations to achieve the throughput limit while effectively guaranteeing bandwidth of gold users. Numerical simulations validate the analysis.

The paper is outlined as follows. In Section 2, we briefly review some background knowledge, which will be the basis of our later discussion. In Sections 3 and 4 we discuss NSAD improvement and the implementation of the new bandwidth guarantee schemes. We illustrate the simulation results in Section 5. Concluding remarks are given in Section 6.

2 Background

Suppose all the stations are in the same hotspot network. All the stations of the system are reachable to each other. A successful data packet transmission will experience the process illustrated in Fig.1, where *t_v* is defined as virtual transmission time of a station (different from the definition in [8]), *t_coll*, *t_free*, *t_succ* and *t_pack* are collision time slots, free time slots, successful transmission of other active mobile stations and data packet transmission time slots of this station, respectively. At any time, the network must be in one of the following 3 states.

State 1 (Under load state): The ratio between *t_coll* and *t_free* is smaller than optimal. The reason is that current contention window is too big when compared with current wireless media load and unnecessary backoff delay is caused. For the parameter set of traditional DCF, this case is unusual.

State 2 (Over load state): In heavy contention scenario, the ratio between *t_coll* and *t_free* is often greater than optimal. The reason is that current contention window is too small when compared with current media load and one successful transmission of data frame will often experience many times of collision. The limited self-adapt capability of the DCF exponential window backoff algorithm cannot tune the access probability of heavy contention stations to achieve optimal.

State 3 (Normal state): In light or moderate contention scenario, the ratio between *t_coll* and *t_free* is around the optimal value, the access probability of the stations is around the optimal and the maximum throughput is achieved.

To avoid the network to evolve into State 1 or State 2, we should carefully design the protocol to make the ratio between *t_free* and *t_coll* approach the optimum, which will lead to the maximum throughput of the wireless LAN. Our algorithm is mainly aimed to solve the problem of State 2. We dynamically tune the initial contention window of DCF according to the load of the network so that the network state will transit from State 2 back to State 3.

Some previous work has been done in the field of network state detection. In [8], the authors' algorithm is based on some statistics obtained by observing the wireless medium. The successful packet transmission of a data packet is confirmed by the successful observation of an ACK packet. The authors of [6] provide each station with a collision counter, which determines how many collisions on average a packet experiences before it is successfully transmitted. Since in the wireless media, we cannot definitely know whether the missing of ACK is caused by wireless media carrier sense fault or collision of data packets, the above 2 mechanisms will both affected by wireless media error or hidden terminal phenomenon. However, from the simulation results of [6] and [8], we can see that the two mechanisms are applicable in the case of low carrier sense fault condition. In this paper, we also use the net state detection mechanism similar to that of [8]. We take the same assumption that any packet without an ACK is a collided packet. So, we can detect the collision length by measuring the length of the transmissions without ACK. Our protocol architecture is depicted in Fig.2.

First, we use our net state detection mechanism to investigate which state the current network is in. Then, we use *NSAD* (New Self-Adapt DCF algorithm) to tune the MAC multi-access of all the ac-

tive stations to achieve theoretical limit. Based on our *NSAD* algorithm, we implement *QoS-NSAD* to support bandwidth guarantee on wireless LANs.

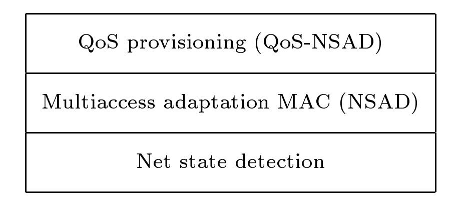


Fig.2. Overview of our new protocol architecture.

3 NSAD Protocol Enhancement

The following notations are introduced to facilitate later contention window mechanism description:

W: Current contention window value;

$W_{\rm init}$: Initial contention window value;

$W_{\rm min}$: Minimum contention window value;

$W_{\rm max}$: Maximum contention window value;

N: The number of active stations (i.e., N stations are contenting the channel);

t_v: Virtual transmission time of an active station (also see Fig.1);

L_p: Time period of sending a data packet with piggybacked initial contention window (illustrated in Fig.3). It equals the time of m successful data packet transmissions of a station, i.e., $L_p = m \times E(t_v)$. We assume $L_p = 40,000$ (slots) in our protocol (shown in Table A1, Appendix);

m: The number of data packets that an active station successfully transmits during L_p;

l: Load factor (also see later definition).

$l_{\rm opt}$: The value of l when throughput achieves maximum. It is considered to be the optimal value. (See [7] for more details).

In order to be self-adapt according to the network contention, we presented a new self-adapt algorithm with the name of *NSAD* (New Self-Adapt DCF algorithm) in [7]. Here, we revisit the *NSAD* mechanism in [7] and propose some modifications in order to further enhance system performance in error-prone or multi-hop environment. The modifications are as follows: a) dynamic m tuning mechanism (m represents how many times a station sends data packet before sending piggybacked $W_{\rm init}$, also see Fig.3) b) *WFM* mechanism, which will be described later in detail. We first introduce a parameter l closely related to the network load in [7]. l is defined as follows:

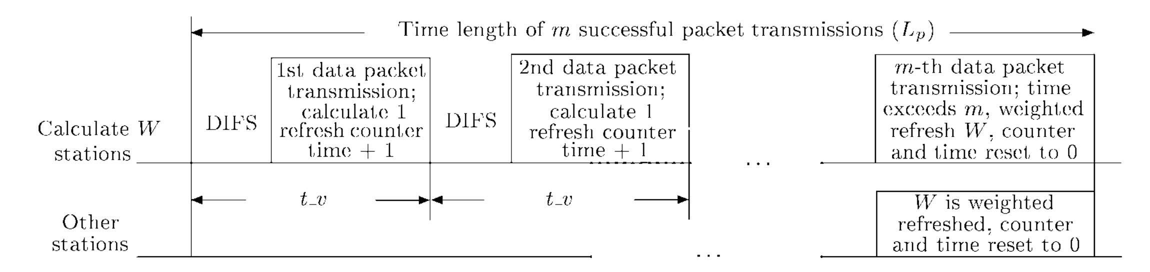


Fig.3. NSAD initial contention window refresh.

Definition. *The wireless link load factor l is defined as the ratio between the mean collision time length and the mean idle time length.*

The stations use a variable *counter* to calculate the times that the measured l is out of the neighborhood of $l_{\rm opt}$. If $|counter|$ is larger than a predefined value in m successful packet transmissions, the station will tune $W_{\rm init}$ and use the transmitting data packet to piggy-back the just tuned initial contention window $W_{\rm init}$ to refresh all the other stations of the single-hop network using weighted refresh mechanism (*WFM*) described later. When the one-hop nearby stations get the piggy-backed window value, they will clear their *t_coll*, *t_free* and *counter* value for next term network contention measurement because their previously measured results are relative to the old initial contention window. The length of the network state measuring periodicity L_p is defined as m times of the successful packet transmissions of a station, i.e., $L_p = m \times t_v$.

In wireless LAN scenario, the newly arrived stations will not have the same initial contention window as the old stations that have been there for a long time. And the wireless medium high bit error rate will prohibit the tuned initial contention window from the sending to all the nearby stations without error. So there will undoubtedly be stations with different initial contention window sizes in the network. Smaller initial contention window stations will have smaller mean t_v length. If all the stations use the same m value, the small initial contention window will have smaller L_p and have

higher priority when refreshing contention window. As a result, W_{init} cannot be tuned to optimal (always less than optimal, which has been proved true by simulation results). So we should tune m value according to current initial contention window size W_{init} in order to guarantee the fairness between all the stations. Large W_{init} stations should have small m value and small W_{init} stations should have large m value so that all the stations have similar L_p value. The m tuning and calculation are shown in the appendix.

The initial contention window piggyback mechanism will flood an error tuned initial contention to other stations if we do not adopt some window broadcast control mechanism. In order to prevent error tuned initial contention window from affecting the network, we adopt the Weighted Refresh Mechanism (*WFM*) when a station receives a packet with piggybacked initial contention window. The philosophy is that any station should be equally affected by the net state detection results of other stations (i.e., the piggy-backed initial contention window) and the results of local net state detection. So, we set the default value $\mu = 0.5$ in the weighted refresh mechanism as described below:

$$W = \mu \cdot W + (1 - \mu) \cdot W_{\text{piggy-back}}. \tag{1}$$

4 Bandwidth Guarantee Schemes

4.1 Gold Users Throughput Analysis

We have the following equations about the channel access probability in a system with two QoS levels (p_g and p_o are the collision probabilities of a gold user and an ordinary user respectively when trying to access the channel, n_g and n_o are contention window backoff times of a gold user and an ordinary user respectively):

$$\begin{cases} \tau_g = 2\sum_{i=0}^{n'} p_g^i \Big/ \Big[\sum_{i=0}^{n'} p_g^i + (W_g + 1) \sum_{i=0}^{n_g} (2p_g)^i \\ \qquad + (W_{\max} + 1) \sum_{i=n_g+1}^{n'} p_g^i \Big] \\ \tau_0 = \dfrac{\sqrt{[N + 2(N-1)(T_c^* - 1)]/N} - 1}{(N-1)(T_c^* - 1)} \\ \quad \approx \dfrac{1}{N\sqrt{T_c^*/2}} \\ n_g = \log_2[(W_{\max} + 1)/(W_g + 1)] \\ n_o = \log_2[(W_{\max} + 1)/(W_o + 1)]. \end{cases} \tag{2}$$

Note $k = \tau_g/\tau_0$, thus:

$$k = N \cdot \sqrt{2T_c^*} \cdot \Big(\sum_{i=0}^{n'} p_g^i \Big) \cdot \Big\{ \Big[\sum_{i=0}^{n'} p_g^i + (W_g + 1) \sum_{i=0}^{n_g} (2p_g)^i + (W_{\max} + 1) \sum_{i=n_g+1}^{n'} p_g^i \Big] \Big\}^{-1}. \tag{3}$$

To simplify the analysis, one gold user is regarded as k virtual ordinary users, and the total system can be regarded as $N = k * N_g + N_o$ users, thus

$$p_g = 1 - (1 - \tau_o)^{N-k} \tag{4}$$

$$p_o = 1 - (1 - \tau_o)^{N-1}. \tag{5}$$

Using (2) to calculate n_g, noting $x^{\log_z y} = y^{\log_z x}$, after some simplifications, we have

$$k = \Big(N \cdot \sqrt{2T_c^*} \cdot \sum_{i=0}^{n'} p_g^i \Big) \cdot \Big\{ (W_g + 1) \cdot \sum_{i=0}^{n'} (2p_g)^i \cdot \Big[\frac{1}{1 - (2p)^{n'+1}} - \Big(\frac{W_g + 1}{W_{\max} + 1} \Big)^{\log_2(\frac{1}{2p_g})} \cdot \frac{p_g}{(1 - p_g)[1 - (2p_g)^{n'+1}]} \Big] + \sum_{i=0}^{n'} p_g^i - \frac{p_g^{n'+1}}{1 - p_g} \cdot (W_{\max} + 1) \Big\}^{-1}. \tag{6}$$

In heavy contention condition, priority users will tune W_g to maximum $\alpha * (W_{\min} + 1) - 1$. So:

$$k = \Big(N \cdot \sqrt{2T_c^*} \cdot \sum_{i=0}^{n'} p_g^i \Big) \cdot \Big\{ \alpha (W_{\min} + 1) \cdot \sum_{i=0}^{n'} (2p_g)^i \cdot \Big[\frac{1}{1 - (2p_g)^{n'+1}} - \Big(\alpha \frac{W_{\min} + 1}{W_{\max} + 1} \Big)^{\log_2(\frac{1}{2p_g})} \cdot \frac{p_g}{(1 - p_g)[1 - (2p_g)^{n'+1}]} \Big] + \sum_{i=0}^{n'} p_g^i - \frac{p_g^{n'+1}}{1 - p_g} \cdot (W_{\max} + 1) \Big\}^{-1}. \tag{7}$$

Here we get the relationship between the number of ordinary users N (one gold user is represented as k ordinary users) and k value. Note that $p_o = 1 - e^{-\frac{1}{\sqrt{T_c^*/2}}}$ in heavy contention condition[9]. When using RTS/CTS option in DSSS 2Mbps condition, RTS collision length $T_c^* = 29.0$ (slots). Together with (4), we get $p_g < p_o = 0.23 < 1/4$. In

the case of data packet collision, the collision length will be longer and p_o will be still smaller. Note that $n' = 7$, thus $p_g^{n'+1}$ is far less than 1. Obviously, the absolute value of last 2 terms in the denominator of (7) is far less than the first term and can be ignored. Thus

$$k \approx \Big(N \cdot \sqrt{2T_c^*}\frac{1-2p_g}{1-p_g}\Big)\Big\{\alpha(W_{\min}+1) \cdot \Big[1-\Big(\alpha\frac{W_{\min}+1}{W_{\max}+1}\Big)^{\log_2(\frac{1}{2p_g})} \cdot \frac{p_g}{(1-p_g)}\Big]\Big\}^{-1} = \theta N \tag{8}$$

$$\theta = \Big(\sqrt{2T_c^*}\frac{1-2p_g}{1-p_g}\Big) \cdot \Big\{\alpha(W_{\min}+1) \cdot \Big[1-\Big(\alpha\frac{W_{\min}+1}{W_{\max}+1}\Big)^{\log_2(\frac{1}{2p_g})} \cdot \frac{p_g}{(1-p_g)}\Big]\Big\}^{-1} \approx \Big(\sqrt{2T_c^*}\frac{1-2p_g}{1-p_g}\Big) \cdot [\alpha(W_{\min}+1)]^{-1} \quad (\alpha \cdot (W_{\min}+1) = W_{\max}+1) \tag{9}$$

According to (9), when α is a small value (i.e., when $\alpha \cdot (W_{\min}+1) = W_{\max}+1$), θ is approximately inversely proportional to α. At the same time, suppose N is big enough, then

$$p_g = 1-(1-\tau_o)^{N_o+kN_g-k} = 1-(1-\tau_o)^{(1-\theta)N} \approx 1-e^{-\frac{1-\theta}{\sqrt{T_c^*/2}}} \tag{10}$$

(9) and (10) form a nonlinear system of 2 unknowns θ and p_g and we can use numerical methods to solve them. Let $W_{\max} = 1{,}023$, $W_{\min} = 31$, $\alpha = 2.0, T_c^* = 29.0$ (slots), we can get $\theta = 0.0867$, $p_g = 0.213$.

Let S be the system total throughput. The bandwidth which a gold user can obtain is S_g. Suppose that all users have similar data packet length, then we can get:

$$\frac{S_g}{S} = [\tau_g(1-p_g)] \cdot \Big[N_o\tau_o(1-\tau_o)^{N_o+kN_g-1} + kN_g\tau_o(1-\tau_o)^{N_o+kN_g-k}\Big]^{-1}$$

Together with (10), noting that $e^{-\theta/\sqrt{T_c^*/2}} \approx 1$, we get

$$S_g = \frac{1}{e^{-\theta/\sqrt{T_c^*/2}} \cdot \Big(\frac{1}{\theta}-N_g\Big)+N_g} \cdot S \approx \theta \cdot S = 0.0867S \tag{11}$$

From (11) we can see the two excellent properties of the *QoS-NSAD* algorithm in heavy contention environment.

First, the bandwidth of gold users is well guaranteed to be a definite share of total bandwidth and will not degrade with the growing number of active ordinary users.

Second, we can easily tune initial contention window upper bound parameter α in order to change the bandwidth that a gold user can achieve.

4.2 Implementation Details

Practically, the *QoS-NSAD* algorithm will have the following modification based on *NSAD* algorithm:

1) The gold user's initial contention window is confined in $[W_{\min}, \alpha \times (W_{\min}+1)-1]$, $(\alpha \times (W_{\min}+1)-1 < W_{\max})$, while the ordinary user's initial contention window is tuned in the range of $[W_{\min}, W_{\max}]$ just as in *NSAD*.

2) In order to prevent the priority stations from piggy-backing their small initial contention window to best-effort stations, priority stations set their m value to 4 times of the best-effort stations' so that the best-effort stations have priority in exchanging initial contention window. When the priority stations receive the large contention window (larger than $\alpha \times (W_{\min}+1)-1$) piggybacked by packets of best-effort stations they only tune their initial contention window to the upper bound, i.e., $\alpha \times (W_{\min}+1)-1$.

5 Simulation Analysis

To validate our algorithm in previous sections, we use the $ns-2$[10] *(Network Simulator)* of Berkeley University to set up the simulation model.

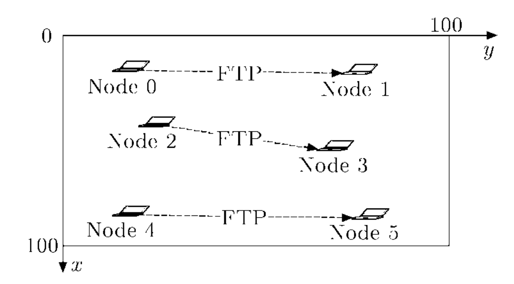


Fig.4. Single-hop simulation topology.

We use the hot-spot cells scenario shown in Fig.4. In order to simplify the simulation model, we have the following hypotheses.

1) We just care for the multi-access aspect of the new algorithm, so we do not take *hidden terminal*

and *exposed terminal* into consideration. Though *hidden terminal* is quite usual in real world case, our basic idea is that we only use the algorithm to those that can be seen to each other.

2) The buffer is large enough for stations and the loss of frame is all due to collision and timeout, which can be easily achieved for stations.

The topology is $N/2$ communication pairs (we just draw 3 node pairs in the figure). The N nodes are uniformly distributed in the 100×100 topology. All the nodes are in the communication range. We set up TCP connection between the odd node and even node for FTP applications. The version of TCP is NewReno. Important parameters are listed in Table 1. Nodes 0—7 are 4 priority station pairs; the other stations are all best-effort stations. We set $\alpha = 2$ in the simulation. In order to show the effect of *QoS-NSAD*, we compare *QoS-NSAD* with the contention window differentiation algorithm that is based on DCF[11] (mentioned as *QoS-DCF* in this paper). In *QoS-DCF*, priority users' contention window is set to be in the range of $[W_{\min}, (W_{\max}-1)/2]$, while best-effort users' is in the range of $[(W_{\min}-1) \times 2, W_{\max}]$. We compared goodput and MAC delay of the 2 algorithms.

Table 1. Important Simulation Parameters

Channel bit rate	2Mbps
Propagation model	Shadowing
Packet error rate	10% at 150 meters far
Communication pairs	5, 15, 25, 35, 50
$W_{\min}$, $W_{\max}$	31, 1,023
FTP start and stop time	10.0s, 135.0s
Estimated l_{opt}	0.85 (using RTS/CTS)
W_{init} refresh threshold σ	0.3
Data packet length	1,500 bytes
μ	0.5

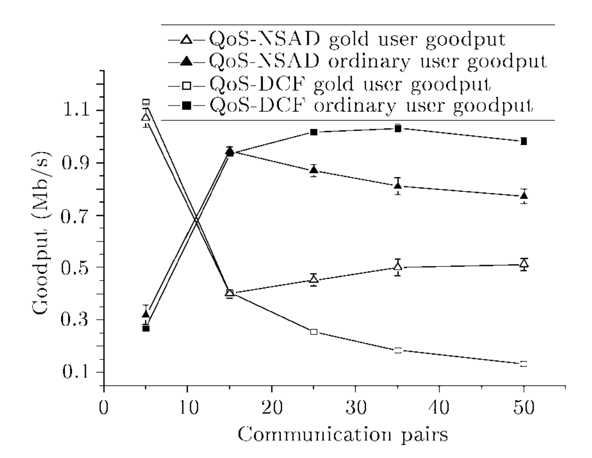


Fig.5. User goodput comparison of *QoS-DCF* and *QoS-NSAD* algorithms.

5.1 Goodput

In Fig.5, we compare bandwidth guarantee character of the 2 different MAC algorithms. When using *QoS-DCF*, with the increasing number of communication pairs, the goodput of priority stations monotonically decreases. When we use *QoS-NSAD*, the total goodput of priority stations remains between 0.40 and 0.50 in heavy contention environment (when node number is equal to or larger than 30). From (15), we know that the total goodput of gold users can be calculated as: $0.0867 \times 4 \times 1.3 = 0.45$, where 1.3 is the approximate total TCP goodput of *QoS-NSAD* mechanism in the simulation results of Fig.6.

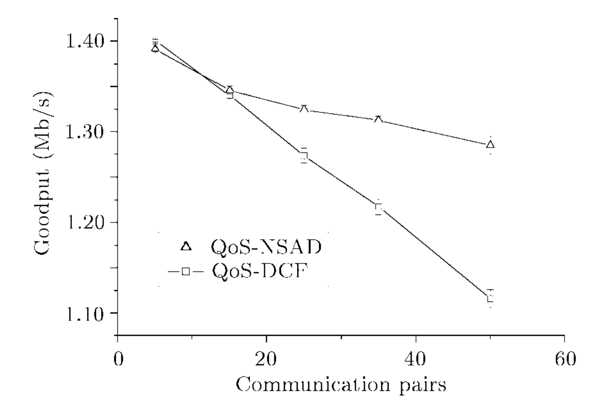


Fig.6. System total goodput comparison of *QoS-DCF* and *QoS-NSAD* algorithms.

In Fig.6, we can see the total system goodput of the two MAC algorithms. We can see that *QoS-NSAD* keeps the self-adapt feature of *NSAD* and achieves larger goodput in heavy contention condition.

5.2 MAC Delay

From Fig.7, we can see that when using the *QoS-NSAD* algorithm, the MAC delay of gold users is well guaranteed. The average delay of gold users remains less than 0.1 second even when there are 50 communication pairs contending the channel. However, if we use the *QoS-DCF* algorithm, the average delay of gold users has an increasing trend with the growing number of active communication pairs. We have also compared the MAC delay of ordinary users in Fig.7. In heavy contention condition, ordinary users in *QoS-NSAD* algorithm will tune the initial contention window to a large value. As a result, the MAC delay is prolonged, which is shown very clearly in Fig.7. In light contention condition, the 2 algorithms have similar ordinary user

MAC delay. Note that using *QoS-DCF* algorithm the MAC delay of gold users is almost the same as ordinary users in both light contention condition and heavy contention condition.

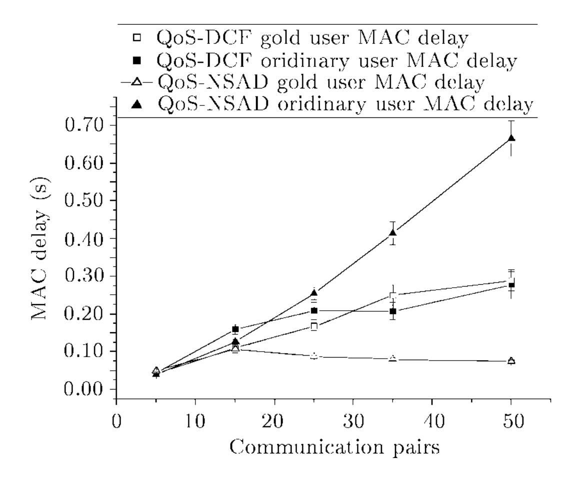


Fig.7. User MAC average delay comparison of *QoS-DCF* and *QoS-NSAD* algorithms.

6 Conclusion

In this paper, we have presented a new MAC architecture to provide self-adapt channel access on the basis of IEEE 802.11 DCF protocol. Some improvements of our previously proposed *NSAD* algorithm are proposed first. Then a new *NSAD*-based MAC mechanism that can support bandwidth guarantee is proposed (*QoS-NSAD*). We have then analyzed the priority stations' bandwidth in heavy contention environment when using *QoS-NSAD*. Lastly, simulation results have validated our new bandwidth guarantee schemes and the theoretical analysis.

References

[1] IEEE 802.11, Wireless LAN Medium Access Control (MAC) and Physical (PHY) Layer Specifications, 1999.

[2] Yong Peng, Haitao Wu, Keping Long, Shiduan Cheng. Simulation analysis of TCP performance on IEEE 802.11 wireless LAN. In *Proc. 2001 Int. Conf. Info-Tech and Info-Net,* Beijing, Oct. 29–Nov. 1, 2001, 2: 520–525.

[3] Imad Aad, Claude Castelluccia. Differentiation mechanisms for IEEE 802.11, INFOCOM 2001. *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proc. IEEE,* April 22–26, 2001, 1: 209–218.

[4] Andras Veres, Andrew T Campbell, Michael Barry, Li-Hsiang Sun. Supporting service differentiation in wireless packet networks using distributed control. *IEEE JSAC*, October 2001, 19(10): 2081–2093.

[5] Shugong Xu, Tarek Saadawi. Revealing TCP incompatibility problem in 802.11-based wireless multi-hop networks. *Global Telecommunications Conference,* IEEE, Nov. 25–29 2001, 5: 2847–2851.

[6] Albert Banchs, Xavier P'erez. Providing throughput guarantees in IEEE 802.11 wireless LAN. *Wireless Communications and Networking Conference,* IEEE, March 17–21, 2002, 1: 130–138.

[7] Yong Peng, Haitao Wu, Shiduan Cheng, Keping Long. A new self-adapt DCF algorithm. *Global Telecommunications Conference,* IEEE, Nov. 17–21, 2002, pp.87–91.

[8] Cali F, Conti M, Gregori E. IEEE 802.11 protocol: Design and performance evaluation of an adaptive backoff mechanism. *IEEE JSAC*, 2000, 18(9).

[9] Giuseppe Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE JSAC,* March 2000, 18(3).

[10] The Network Simulator. http://www.isi.edu/nsnam/ns/.

[11] Seung-Seok Kang, Mutka M W. Provisioning service differentiation in ad hoc networks by modification of the backoff algorithm. In *Proc. Tenth International Conference on Computer Communications and Networks,* Oct. 15–17, 2001, pp.577–580.



Yong Peng was born in 1978 and received his B.S. degree in computer science from the Beijing Univ. Posts & Telecommunications (BUPT) in 1999. He is currently a Ph.D. candidate of the National Laboratory of Switching Technology and Telecommunication Networks at BUPT. His research interests are QoS and wireless networks.

Shi-Duan Cheng is a professor and Ph.D. supervisor. She graduated from BUPT in 1963. She has published more than 70 papers and several books in the field of telecommunications. Her research interests cover ISDN, ATM, protocol engineering, network performance, security and survivability.

Appendix. Dynamic Tuning m According to W_{init}

A station encounters collision with the probability of $p = 1 - (1-\tau)^{N-1}$. From [9]:

$$\tau = \tau_{\text{opt}} = \frac{\sqrt{[N + 2(N-1)(T_c^* - 1)]/N} - 1}{(N-1)(T_c^* - 1)} \approx \frac{1}{N\sqrt{T_c^*/2}} \quad \text{(A1)}$$

Using the limit equation $\lim\limits_{x\to\infty}\left(1+\dfrac{1}{x}\right)^x = e$, when $N \gg 1$, we have:

$$p = 1 - (1-\tau_{\text{opt}})^{N-1} \approx 1 - e^{-1/\sqrt{T_c^*/2}} \quad \text{(A2)}$$

At the same time, $t_v = t_coll + t_free + t_pack + t_succ$, thus

$$E(t_v) = E(t_coll) + E(t_free) + E(t_pack) + E(t_succ) \quad \text{(A3)}$$

$$\frac{E(t_coll)}{E(t_free)} = T_c^* \frac{1-(1-\tau)^N - N\tau(1-\tau)^{N-1}}{(1-\tau)^N} \quad \text{(A4)}$$

$$E(t_coll) \approx (e^{1/\sqrt{T_c^*/2}} - 1 - 1/\sqrt{T_c^*/2})E(t_free)T_c^*$$

$$\frac{[E(t_succ) + E(t_pack)]}{E(t_free)} = \frac{N\tau(1-\tau)^{N-1}}{(1-\tau)^N} \cdot t_pack$$

Thus

$$[E(t_succ) + E(t_pack)] = E(t_free)\frac{N\tau}{(1-\tau)} \cdot t_pack \approx E(t_free) \cdot t_pack \cdot \frac{1}{\sqrt{T_c^*/2}} \quad \text{(A5)}$$

$E(t_free)$ can be calculated as (A6).

$$\begin{aligned} E(t_free) =& \sum_{i=0}^{n-1}\Big\{p^i \cdot (1-p) \cdot \sum_{j=0}^{i}\Big[\sum_{k=0}^{2^j(W_{\text{init}}+1)-1}\frac{k}{2^j(W_{\text{init}}+1)}\Big]\Big\} \\ &+ \sum_{i=n}^{n'}\Big\{p^i \cdot (1-p) \cdot \Big\{\sum_{j=0}^{n-1}\Big[\sum_{k=0}^{2^j(W_{\text{init}}+1)-1}\frac{k}{2^j(W_{\text{init}}+1)}\Big] + \sum_{j=n}^{i}\Big(\sum_{k=0}^{W_{\max}}\frac{k}{W_{\max}+1}\Big)\Big\}\Big\} \\ =& \Big\{\frac{[1-(2p)^n](1-p)}{1-2p} - \frac{(1-p^n)}{2}\Big\} \cdot (W_{\text{init}}+1) - \frac{1-p^n}{2(1-p)} + \frac{np^n}{2} \\ &+ \frac{(p^n - p^{n'+1})}{2}\Big[(2^n-1)(W_{\text{init}}+1) - n - (n-1)W_{\max} + \frac{W_{\max}}{1-p}\Big] \\ &+ \frac{W_{\max}}{2}[p^n(n-1) - p^{n'+1}n'] \\ =& \frac{(W_{\text{init}}+1)}{2}\Big\{\frac{2(1-p)[1-(2p)^n]}{1-2p} + (p^n-1) + (p^n - p^{n'+1}) \cdot (2^n-1)\Big\} \\ &- \frac{1-p^n}{2(1-p)} + \frac{np^{n'+1}}{2} + \frac{W_{\max}}{2} \cdot \Big[p^{n'+1}(n-n'+1) + \frac{(p^n - p^{n'+1})}{1-p}\Big] \end{aligned} \quad \text{(A6)}$$

In the above calculation of $E(t_free)$, assume that there is no packet loss caused by max retransmission limit when we use the *NSAD-like* tuning algorithm. We take this assumption from the *NSAD* simulation results of packet retransmission loss. For the number of nodes N is big enough, we use (A2) to calculate p. t_pack is a constant value. Using the parameters of DSSS 2Mb/s 1,500 bytes data packet length, t_pack equals the time slots of RTS + SIFS + CTS + SIFS + DATA + SIFS + ACK + DIFS, DATA length is calculated as the mean length of upper layer TCP data packets and ACK packets, i.e., $(1,500 + 40)/2 = 770$ (bytes). (Note that here we ignore other packets such as the routing packets. Practically in a real system, the mean data packet length can be measured online). After some calculation, $t_pack = 206$ (slots). When using RTS option, T_c^* can be calculated as: RTS + EIFS= 29.0 (slots). We take $n' = 7$ as in the standard[1]. We use Table 1 for n according to different W_{init} values. Using (A3)–(A6), we can calculate how $E(t_v)$ varies according to W_{init}. Calculation results are shown in Table A1.

Table A1. Relationship of W_{init}, Mean t_v and m ($L_p = 40,000$ slots = 0.8 second)

W_{init}	$E(t_v)$ (slots)	$m = \frac{L_p}{E(t_v)}$	W_{init}	$E(t_v)$ (slots)	$m = \frac{L_p}{E(t_v)}$
31	1,623	25	543	23,886	2
63	3,258	12	575	24,785	2
95	4,853	8	607	25,684	2
127	6,448	6	639	26,583	2
159	7,954	5	671	27,482	1
191	9,460	4	703	28,381	1
223	10,966	4	735	29,280	1
255	12,472	3	767	30,179	1
287	13,786	3	799	31,078	1
319	15,101	3	831	31,978	1
351	16,415	2	863	32,877	1
383	17,730	2	895	33,776	1
415	19,044	2	927	34,675	1
447	20,358	2	959	35,574	1
479	21,673	2	991	36,473	1
511	22,987	2	1,023	37,372	1

We tune m adaptively according to Table A1. In this way, the fairness between nodes with large W_{init} and those with small W_{init} is achieved.